Noname manuscript No.
(will be inserted by the editor)

Lewis Ntaimo · Suvrajeet Sen

A Branch-and-Cut Algorithm for Two-Stage Stochastic Mixed-Binary Programs with Continuous First-Stage Variables

Received: date / Accepted: date

Abstract This paper presents a branch-and-cut method for two-stage stochastic mixed-integer programming (SMIP) problems with continuous first-stage variables. This method is derived based on disjunctive decomposition (D^2) for SMIP, an approach in which disjunctive programming is used to derive valid inequalities for SMIP. The novelty of the proposed method derives from branching on the first-stage continuous domain while the branch-and-bound process is guided by the disjunction variables in the second-stage. Finite convergence of the algorithm for mixed-binary second-stage is established and a numerical example to illustrate the new method is given.

Keywords stochastic programming · disjunctive decomposition · branch-and-bound · branch-and-cut

1 Introduction

Stochastic mixed-integer programming (SMIP) is a branch of stochastic programming that deals with stochastic programs in which the decision variables involve integrality requirements. SMIP has many applications in operations research and combines two generally difficult classes of problems: stochastic programs and integer programs. Therefore, by inheriting the properties of both generally difficult classes of problems, SMIP is among the most challenging classes of optimization problems. In two-stage SMIP with recourse, the first-stage decisions have to be made "here and now" in the face of future uncertainty in the problem data, while the second-stage (recourse) decisions

L. Ntaimo: Department of Industrial and Systems Engineering, Texas A&M University, 3131 TAMU, College Station, TX 77843, USA. E-mail: ntaimo@tamu.edu

S. Sen: Department of Systems and Industrial Engineering, The University of Arizona, P.O. Box 210020, Tucson, Arizona 85721, USA. E-mail: sen@sie.arizona.edu

are determined based on the realization of the random outcomes. In this paper we consider the following two-stage SMIP problem:

$$\min_{x \in X} c^\top x + E[f(x, \tilde{\omega})], \tag{1}$$

where c is a known vector in $\Re^{n_1}$, $X \subseteq \Re^{n_1}$ is a set of feasible first-stage decisions, and $E[.]$ is the usual mathematical expectation operator with

$$E[f(x, \tilde{\omega})] = \sum_{\omega \in \Omega} p_\omega f(x, \omega),$$

$\tilde{\omega}$ is a multi-variate discrete random variable with a realization (scenario) ω with probability p_ω and sample space Ω. For any ω,

$$f(x, \omega) = \min \ q(\omega)^\top y, \tag{2a}$$
$$\text{s.t. } Wy \geq r(\omega) - T(\omega)x, \tag{2b}$$
$$y \geq 0, \ y_j \text{ binary}, \ j \in J_2. \tag{2c}$$

In problem formulation (2), $q(\omega)$ is the cost vector in $\Re^{n_2}$ for scenario $\omega \in \Omega$ and J_2 is an index set that may include some or all the variables listed in $y \in \Re^{n_2}$. We consider instances of problem (1-2) under the following assumptions:

(A1) Ω is a finite set.
(A2) $X = \{x \in \Re^{n_1}_+ \mid Ax \geq b\}$.
(A3) $f(x, \omega) < \infty$ for all $(x, \omega) \in X \times \Omega$.

Assumption (A3) requires that the subproblem (2) remain feasible for all $(x, \omega) \in X \times \Omega$, a property referred to as relatively complete (integer) recourse (Wets 1974).

When the second-stage involves continuous variables only, the second-stage objective function (recourse function) is a well-behaved piecewise linear and convex function of the first-stage variables. Therefore, Benders' decomposition [3] is applicable. However, when integrality restrictions appear in the second stage, computational challenges arise. The recourse function is now lower semicontinuous with respect to the first-variables [5], making it generally nonconvex [12]. Thus direct Benders approaches are no longer applicable. When the first-stage decisions are pure binary variables and the second-stage decisions involves integrality restrictions, finite termination is justified when the algorithm is based on branching on the first-stage variables. Algorithms dealing with SMIPs with binary first-stage variables include the decomposition algorithm of [9], the D^2 algorithm ([22, 20]) a modified Benders' algorithm for SMIP [18] based on the reformulation-linearization technique or RLT ([15–17]), and decomposition with branch-and-cut (D-BAC) and D^2 with branch-and-cut (D^2-BAC) algorithms [23].

The algorithms developed by [22], [18], and [23] require that $x \in vert(X)$. These algorithms exploit the fact that if the first-stage solution $x \in vert(X)$ (as is the case with pure binary first-stage), then for a given solution $\bar{x} \in vert(X)$ and $\omega \in \Omega$, the extreme points of $conv\{(x, y) : T(\omega)x + Wy \geq$

$r(\omega), y \geq 0, y_j \in \{0,1\}, j \in J_2\} \cap \{(x,y) : x = \bar{x}\}$ have binary values for $y_j, \forall j \in J_2$. However, this requirement no longer necessarily holds for SMIPs with continuous first-stage variables. Thus the case of SMIPs with continuous first-stage variables is much more challenging, and the aforementioned algorithms are inapplicable. Furthermore, very few algorithms have been developed for this class of SMIPs and alternative approaches are needed. In this paper we propose a novel branch-and-cut (BAC) method for SMIP with continuous first-stage variables. Branch-and-bound in this method involves branching on the first-stage continuous domain while cut generation is done in the second-stage using the D^2 method for SMIP [22]. An earlier version of this paper appears as a chapter in [10].

Algorithms that address problem (1-2) include the BAC algorithm for SMIPs with mixed-binary variables in both stages derived by [6]. This method uses disjunctive programming [2] to derive lift-and-project cuts in the $(x, y(\omega))$-space based on the extensive form or deterministic equivalent problem (DEP) of problem (1-2). [1] derives a branch-and-bound method for SMIPs with general first-stage and pure integer second-stage variables. They use a transformed space in which tender variables $\chi = Tx$ are used to partition the problem using a hyperrectangular partitioning process. Recently, [19] proposed a decomposition-based branch-and-bound (DBAB) algorithm based on a hyperrectangular partitioning process on the first-stage continuous domain. They follow a modified Benders' decomposition approach in which the subproblems define lower bounding value functions of the first-stage variables. The subproblems are derived by sequentially constructing a partial convex hull representation of the two-stage solution space using the RLT technique. For surveys on SMIP we refer the reader to [13], [8], and [21].

The rest of this paper is organized as follows. In the next section we give some preliminaries on disjunctive decomposition for SMIP. In Section 3 we derive the new algorithm and provide a formal outline of the algorithm in Section 4. A detailed numerical example to illustrate the new approach is given in Section 5. We end the paper with some concluding remarks in Section 6.

2 Preliminaries

The theory for D^2 for SMIP is derived in [22] and illustrated in [20]. Computational results with the D^2 method are reported in [11]. The D^2 method avoids solving scenario subproblem MIPs at every iteration of the algorithm but instead, performs a sequential convexification process that involves solving scenario subproblem LP relaxations. MIP solves are done only when necessary, such as when computing upper bounds for the problem. Sequential convexification is achieved via the common-cut-coefficients (C^3) theorem [20], which allows for a cut (referred to as a D^2 cut) generated for one scenario subproblem to be easily translated into a cut that is valid for another scenario subproblem. The D^2 cut has the form $\pi^\top y \geq \pi_c(x, \omega)$, where π is the common-cut-coefficient vector and the righthand side is a convexification of the function $\pi_0(x, \omega)$. For a given ω the function $\pi_c(x, \omega)$ is a linear function

of x, while the function $\pi_0(x,\omega)$ is a piecewise linear concave function of x [22].

Both π and $\pi_0(x,\omega)$ are generated by forming and solving a simple recourse problem (problem 18 in [22]), which we shall refer to as the C^3-SLP. The solution of this SLP provides the π as well as the multipliers λ_0, and λ_1 associated with the disjunctions used in cut formation and define $\pi_0(x,\omega)$ for each $\omega \in \Omega$. The convexification of $\pi_0(x,\omega)$ to $\pi_c(x,\omega)$ is achieved by forming and solving an LP (problem 19 in [22]) for each $\omega \in \Omega$. This LP, we shall refer to as RHS-LP, is derived via a strategy from reverse convex programming in which disjunctive programming is used to provide facets of the convex hull of reverse convex sets [14]. Consequently, the LP relaxation of (2) in the K-th algorithmic iteration of the D^2 algorithm has the form

$$f_c^K(x,\omega) = \min\ q^\top y, \tag{3a}$$
$$\text{s.t. } Wy \geq r(\omega) - T(\omega)x, \tag{3b}$$
$$(\pi^k)^\top y \geq \pi_c^k(x,\omega),\ k \in \Theta_K, \tag{3c}$$
$$y \geq 0, \tag{3d}$$

for all $\omega \in \Omega$. The set Θ_K is the set of algorithmic indices k at which a D^2 cut is generated. The scenario subproblem LP (3) can be rewritten in compact form as follows:

$$f_c^K(x,\omega) = \min\ q^\top y, \tag{4a}$$
$$\text{s.t. } W^K y \geq \rho^K(x,\omega), \tag{4b}$$
$$y \geq 0. \tag{4c}$$

where, W^K is a result of augmenting the recourse matrix W with $\{(\pi^k)^\top\}_{k\in\Theta_K}$ and $\rho^K(x,\omega)$ is the result of augmenting the righthand side vector $r(\omega) - T(\omega)x$ with $\{\pi_c^k(x,\omega)\}_{k\in\Theta_K}$.

In order to set the stage for the derivation of the proposed method, we first state the theorem that addresses the identification of an optimal solution to the SMIP problem under the requirement that $x \in vert(X)$. SMIP problems with purely (0-1) binary first-stage decision variables satisfy this requirement. The theorem provides a starting point for the derivation of a convexification process for the continuous first-stage case.

Theorem 1 **[22]** *Suppose that $X = \{x \in \Re_+^{n_1} \mid Ax \geq b\}$ includes the constraints $-x \geq -1$, and supposed that assumptions A1-A3 hold. Suppose the D^2 algorithm identifies extreme point solutions of the C^3-SLP (problem 18 in [22]. Then $\exists \bar{K} < \infty$ such that for all $k > \bar{K}$, $f_c^k(x^k,\omega) = f(x^k,\omega)$ for all $\omega \in \Omega$ whenever $x^k \in vert(X)$.*

Proof See [22]. ■

Theorem 1 shows that the D^2 method generates all valid inequalities for any second-stage MIP after finitely many iterations due to the facial property of the subproblem. Hence for any $x^k \in vert(X), \pi_c(x^k,\omega) = \pi_0(x^k,\omega)$. This

property does not necessary hold for $x^k \notin vert(X)$ as is the case with SMIP problems with continuous first-stage variables. The *violation* of this property will guide us towards the derivation of the new BAC method for the continuous first-stage case.

3 The Branch-and-Bound Approach

Consider an iteration K of the D^2 algorithm applied to problem (1-2). Then the algorithm has iteratively identified $\{\pi^k, \pi_0^k(x,\omega), \pi_c^k(x,\omega)\}_{k\in\Theta_K}$ and the master program has the following form:

$$\min\ c^\top x + \eta \tag{5a}$$
$$\text{s.t. } Ax \geq b, \tag{5b}$$
$$\beta_k{}^\top x + \eta \geq \alpha_k,\ k = 1, ..., K, \tag{5c}$$
$$x \geq 0. \tag{5d}$$

The variable η provides a piecewise linear approximation of the subproblem expected objective function $E[f(x,\tilde{\omega})]$. Constraints (5c) are the Benders-type [3] optimality cuts, and constraint (5d) imposes the nonnegativity requirements on x. In constraint (5c) the righthand side $\alpha_k = E[\psi^k(\tilde{\omega})^\top r^k(\tilde{\omega})]$ and $\beta_k = E[\psi^k(\tilde{\omega})^\top T^k(\tilde{\omega})]$ for $k = 1, ..., K$, where $\psi^k(\omega)$ denotes an appropriately dimensioned vector of optimal dual multipliers associated with constraints (4b) for scenario $\omega \in \Omega$ in iteration k. For each $k \in \Theta_K$ we have a pair $\{\pi_0^k(x,\omega), \pi_c^k(x,\omega)\}$, where

$$\pi_0^k(x,\omega) = \min\{\bar{\nu}_0^k(\omega) - \bar{\gamma}_0^k(\omega)^\top x,\ \bar{\nu}_1^k(\omega) - \bar{\gamma}_1^k(\omega)^\top x\}$$

and from the optimal solution of the RHS-LP (problem 19 in [22]) we obtain

$$\pi_c^k(x,\omega) = \nu^k(\omega) - \gamma^k(\omega)^\top x.$$

Since x is continuous it follows that x is not necessarily an extreme point of X and therefore, the condition $\pi_c(x,\omega) = \pi_0(x,\omega)$ may not hold. Consequently, the D^2 algorithm may not converge to an optimal solution of problem (1-2). Therefore, the proposed approach will involve recording the pairs $\{\pi_0^k(x,\omega), \pi_c^k(x,\omega)\}$ during the execution of the algorithm and then using these data to sequentially identify those pairs that violate the condition $\pi_c^k(x,\omega) = \pi_0^k(x,\omega)$ for a given ω and k. The goal is to create a branch-and-bound tree by partitioning the first-stage continuous feasible set X based on the violated (ω, k) pairs.

An issue in applying a branch-and-bound algorithm over a continuous domain is that the resulting approach may not be finite. Our approach will however be guided by *disjunction variables* in the second-stage, whose total number is finite. Consequently, we will have finitely many subdivisions of X to consider, leading to the finiteness of the branch-and-bound method. Figure 1 gives an illustration of the epigraph of the functions $\pi_c(x,\omega)$ and $\pi_0(x,\omega)$ for some scenario $\bar{\omega} \in \Omega$. In the figure $X = \{l \leq x \leq u\}$, where l and u are lower and upper bounds on x in one dimensional space. Note

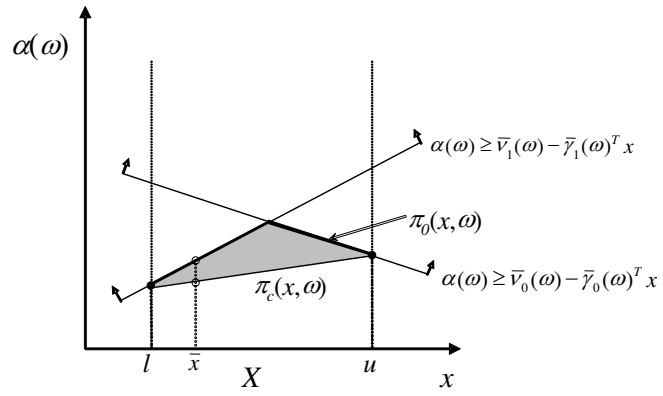


Fig. 1 The epigraph of the functions $\pi_c(x,\omega)$ and $\pi_0(x,\omega)$

that for $x \in vert(X), \pi_c(x,\omega) = \pi_0(x,\omega)$ while $\pi_c(x^k,\omega) < \pi_0(x^k,\omega)$ for some $\bar{x} \notin vert(X)$. Given some first-stage solution $\bar{x} \notin vert(X)$, we can identify a scenario $\varpi \in \Omega$ and an iteration index κ that violates the condition $\pi_0^k(\bar{x},\omega) = \pi_c^k(\bar{x},\omega)$. We propose finding a pair (ϖ,κ) for which the condition is both violated and scenario ϖ has a high probability of occurrence p_ω. This can be accomplished using the following criterion:

$$(\varpi,\kappa) \in \underset{\omega\in\Omega,\ k\in\Theta_K}{\operatorname{argmax}} \left\{p_\omega \left(\pi_0^k(\bar{x},\omega) - \pi_c^k(\bar{x},\omega)\right)\right\} \tag{6}$$

Proposition 1 *Let* $\bar{\gamma}_{q0}^\kappa(\varpi) = \bar{\gamma}_1^\kappa(\varpi) - \bar{\gamma}_0^\kappa(\varpi)$, $\bar{\nu}_{q0}^\kappa(\varpi) = \bar{\nu}_1^\kappa(\varpi) - \bar{\nu}_0^\kappa(\varpi)$, *and* $\bar{\gamma}_{q1}^\kappa(\varpi) = -\bar{\gamma}_{q0}^\kappa(\varpi)$ *and* $\bar{\nu}_{q1}^\kappa(\varpi) = -\bar{\nu}_{q0}^\kappa(\varpi)$. *Also let* $X_{q0} = \{x \mid \bar{\gamma}_{q0}^\kappa(\varpi)^\top x \geq \bar{\nu}_{q0}^\kappa(\varpi), x \geq 0\}$ *and* $X_{q1} = \{x \mid \bar{\gamma}_{q1}^\kappa(\varpi)^\top x \geq \bar{\nu}_{q1}^\kappa(\varpi), x \geq 0\}$. *Suppose that* X^q *denotes some subset of* X. *Then* X^q *can be given as* $X^q = \mathcal{P}_{q0} \cup \mathcal{P}_{q1}$, *where*

$$\mathcal{P}_{q0} = X_q \cap X_{q0} \tag{7}$$

and

$$\mathcal{P}_{q1} = X_q \cap X_{q1} \tag{8}$$

Proof Condition (6) identifies a pair (ϖ,κ) such that $\pi_0^\kappa(\bar{x},\varpi) > \pi_c^\kappa(\bar{x},\varpi)$. Since

$$\pi_0^\kappa(x,\varpi) = \min\{\bar{\nu}_0^\kappa(\varpi) - \bar{\gamma}_0^\kappa(\varpi)^\top x,\ \bar{\nu}_1^\kappa(\varpi) - \bar{\gamma}_1^\kappa(\varpi)^\top x\} \tag{9}$$

is a piecewise linear concave function of x for a given ϖ, it follows that either

$$\bar{\nu}_0^\kappa(\varpi) - \bar{\gamma}_0^\kappa(\varpi)^\top x \geq \bar{\nu}_1^\kappa(\varpi) - \bar{\gamma}_1^\kappa(\varpi)^\top x \tag{10}$$

or

$$\bar{\nu}_0^\kappa(\varpi) - \bar{\gamma}_0^\kappa(\varpi)^\top x \leq \bar{\nu}_1^\kappa(\varpi) - \bar{\gamma}_1^\kappa(\varpi)^\top x. \tag{11}$$

By intersecting the half-space defined by (10) with X_q and that defined by (11) with X_q the result follows. ■

Proposition 1 allows us to divide X into two subsets as illustrated in Figure 2. Therefore, optimization of the original problem can be carried out over each subset. This will enable us to devise a branch-and-bound procedure for solving problem (1-2) by further subdividing each resulting subset if necessary using the branching constraints (10) and (11). Let Q denote the set of nodes of the branch-and-bound tree for the first-stage and let $q \in Q$ denote a node of the branch-and-bound tree. We initialize branch-and-bound with the initial master problem (5) at the root node of the branch-and-bound tree and the scenario subproblem LP relaxations. We then apply the D^2 algorithm to this problem and after each iteration k we check the condition $\pi_0^k(x^k, \omega) = \pi_c^k(x^k, \omega)$. If it is violated we perform branching by invoking Proposition 1 and generating branching constraints (10) and (11). By branching at node q we create two *sibling nodes* q_0 and q_1 from the *parent node* q, and for each sibling node we have the parent master program with the corresponding branching constraint added, resulting in the set $\mathcal{P}_{q_h}$ of the first-stage feasible set, where , $h \in H$ and $H = \{0, 1\}$. The D^2 algorithm can then be applied to each *sibling node* and the process repeated.

Let k_q denote the algorithmic iteration for the problem at a node $q \in Q$. With q we can associate a sequence of nodes that trace a unique path from the node q to the root node. Let B_q denote this sequence of nodes. For each element $\tau \in B_q$, we have all algorithmic indices $k(\tau)$ as well as indices $\kappa(\tau)$ that identify the iteration κ and scenario ϖ used in the branching process. Thus for each $k \in \kappa(\tau)$ we have a pair (ϖ, κ) with the associated branching constraint coefficients $\bar{\gamma}_{q_h}^k = \bar{\gamma}_{q_h}^\kappa(\varpi)$ and $\bar{\nu}_{q_h}^k = \bar{\nu}_{q_h}^\kappa(\varpi)$ for $h \in H$. Then the master program (5) at a node $q \in Q$ takes the following form:

$$\min\ c^\top x + \eta \tag{12a}$$

$$\text{s.t. } Ax \geq b, \tag{12b}$$

$$\beta_k{}^\top x + \eta \geq \alpha_k,\ k \in k(\tau),\ \tau \in B_q \tag{12c}$$

$$(\bar{\gamma}_{q_h}^k)^\top x \geq \bar{\nu}_{q_h}^k,\ h \in H,\ k \in \kappa(\tau),\ \tau \in B_q, \tag{12d}$$

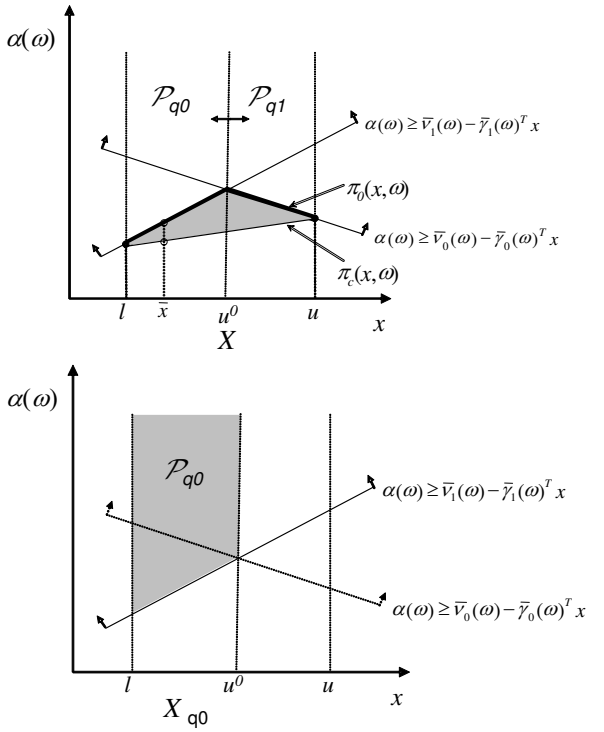
$$x \geq 0. \tag{12e}$$

Constraints (12c) are the Benders's type optimality cuts derived up to iteration k_q for node q. Constraints (12d) are the branching constraints added to the master program for node q. The nodal master program (12) may become infeasible for some node $q \in Q$ beyond the first branch in the branch-and-bound tree. In this case, the node is fathomed, and we backtrack.

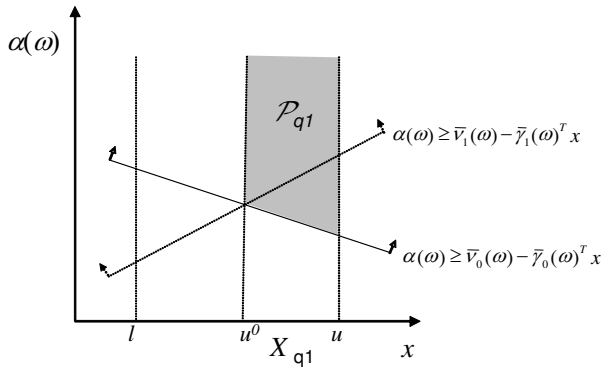
Let us now turn to updating the righthand side for the D^2 cut on which branching is performed. Since branching is carried out based on this D^2 cut, its righthand side $\pi_c^\kappa(x, \varpi)$ must be updated to reflect the branching.

Corollary 1 *Let (ϖ, κ) be defined as in (6). Consider the D^2 cut at which (ϖ, κ) is determined. Then the righthand side $\pi_c^\kappa(x, \varpi)$ for the D^2 cut must be replaced by $\pi_0^\kappa(x, \omega) = \bar{\nu}_1^\kappa(\varpi) - \bar{\gamma}_1^\kappa(\varpi)^\top x$ for the branch $\bar{\gamma}_{q_0}^\kappa(\varpi)^\top x \geq \bar{\nu}_{q_0}^\kappa(\varpi)$, and by $\pi_0^\kappa(x, \omega) = \bar{\nu}_0^\kappa(\varpi) - \bar{\gamma}_0^\kappa(\varpi)^\top x$ for the branch $\bar{\gamma}_{q_1}^\kappa(\varpi)^\top x \geq \bar{\nu}_{q_1}^\kappa(\varpi)$.*

Proof From Proposition 1, it follows that for the branch $\bar{\gamma}_{q_0}^\kappa(\varpi)^\top x \geq \bar{\nu}_{q_0}^\kappa(\varpi)$, the righthand side of the D^2 cut must be the minimum of the two affine functions that define $\pi_0^\kappa(x, \varpi)$. Hence $\pi_0^\kappa(x, \varpi) = \bar{\nu}_1^\kappa(\varpi) - \bar{\gamma}_1^\kappa(\varpi)^\top x$. Similarly,



The epigraph of $\pi_0(x, \omega)$ restricted to $\mathcal{P}_{q0}$



The epigraph of $\pi_0(x, \omega)$ restricted to $\mathcal{P}_{q1}$

Fig. 2 Illustration of the Concepts of Proposition 1

the righthand side of the D^2 cut for the other branch is also justified. ■

The master program (12) in iteration k_q provides a first-stage solution x^{k_q} and the scenario subproblem LP relaxation for $\omega \in \Omega$ and node $q \in Q$ takes the form:

$$f_c^{k_q}(x^{k_q}, \omega) = \min \; q^\top y, \tag{13a}$$

$$\text{s.t. } Wy \geq r(\omega) - T(\omega)x^{k_q}, \tag{13b}$$

$$(\pi^k)^\top y \geq \pi_c^k(x^{k_q}, \omega), \; k \in k(\tau) \setminus \kappa(\tau), \; \tau \in B_q \tag{13c}$$

$$(\pi^k)^\top y \geq \pi_0^k(x^{k_q}, \omega), \; k \in \kappa(\tau), \; \tau \in B_q \tag{13d}$$

$$y \geq 0. \tag{13e}$$

Constraints (13c) are the D^2 cuts generated at each node along the path from the root node to node q whose righthand has not been updated, while constraints (13d) have the righthand updated (due to branching). From here on, we shall refer to problem (12-13) as the *nodal problem* P_q for node q, and shall denote by v_q and V_q the lower and upper bounds of the nodal problem, respectively, as determined by the D^2 algorithm. Before we formally state our algorithm, we should point out that the D^2 cuts generated at a node $q \in Q$ are valid for all the descendant nodes of q. However, they are generally not valid for all the other nodes in Q due to the update performed after branching according to Corollary 1.

4 A Branch-and-Cut Algorithm

We shall now present a formal statement of a D^2 with branch-and-cut algorithm for SMIP with continuous first-stage variables, which we refer to as the D^2-CBAC algorithm. The critical operations in the algorithm are italicized and are discussed in the following subsection.

4.1 The D^2-CBAC Algorithm

We use the following notation in the description:
- $\mathcal{L}$: List of open nodal problems
- v: Lower bound on the optimum
- V: Upper bound on the optimum
- v_q: Lower bound on node q problem optimum
- V_q: Upper bound on node q problem optimum
- k: Iteration index for root node problem
- k_q: Iteration index for node q problem
- x^*: Optimal solution.

Basic D^2-CBAC Algorithm

Step 0. Initialization: Let $\epsilon > 0$ and $x^1 \in X$ be given. Set $k \leftarrow 0$ and initialize $V = \infty$, $v = -\infty$, and add the root nodal problem (12-13) to the list $\mathcal{L}$ of open nodal problems.
Step 1. Termination: If $\mathcal{L} = \emptyset$, terminate with solution x^*. Otherwise *select* a nodal problem P_q from the list $\mathcal{L}$ of currently open nodal problems, and set $\mathcal{L} \leftarrow \mathcal{L} \setminus P_q$ and $k \leftarrow k + 1$.
Step 2. Apply the D^2 Algorithm to the Nodal Problem: Apply one iteration of the D^2 algorithm to the nodal problem P_q. Store the multipliers $\{\lambda^k_{0,1}; \lambda^k_{0,2}\}$, and $\{\lambda^k_{1,1}; \lambda^k_{1,2}\}$. These multipliers define $(\bar{\nu}^k_0(\omega), \bar{\gamma}^k_0(\omega))$, $(\bar{\nu}^k_1(\omega), \bar{\gamma}^k_1(\omega))$ and the corresponding $\pi^k_0(x, \omega)$ and $\pi^k_c(x, \omega)$ for all k and all $\omega \in \Omega$. At the end of the D^2 algorithm iteration one of the following must hold:

(i) Nodal master program (12) becomes infeasible;
(ii) $V_{k_q} - v_{k_q} < \epsilon$;
(iii) $V_{k_q} - v_{k_q} \geq \epsilon$;

If condition (i) is true
Fathom this node by infeasibility. Go to Step 1;
else if condition (ii) is true
Update incumbent:
If $V_{k_q} < V$
then $V \leftarrow V_{k_q}$, $v \leftarrow v_{k_q}$ and $x^* \leftarrow x_{k_q}$;
fathom this node by optimality;
fathom the node list, that is, $\mathcal{L} \leftarrow \mathcal{L} \setminus \{P_q \mid v_q \geq V\}$;
else
fathom this node by bound. Go to Step 1;
else if condition (iii) is true
go to Step 3.
Step 3: Branching: Apply Proposition 1 and perform *branching* to create two nodal problems P_{q_0} and P_{q_1} of the form (12 -13) and add to the list $\mathcal{L}$ of open nodal problems, that is, set $\mathcal{L} \leftarrow \mathcal{L} \cup \{P_{q_0}, P_{q_1}\}$. For node *selection* purposes determine and record v_{q_0} and v_{q_1}, the objective value of the corresponding nodal master program (12) after branching. Go to Step 1. —

4.2 Algorithm Convergence

We now prove finite convergence of the D^2-CBAC algorithm.

Lemma 1 *There exists a finite number t such that after t divisions of the first-stage feasible set X, the branch-and-bound process described in Section 3 finds a node that can be fathomed.*

Proof A node is fathomed if we either encounter $V_q - v_q < \epsilon$ or an infeasible nodal master program. Let $Y(x) = \{y \mid Wy \geq r(\omega) - T(\omega)x, y \geq 0, -y_j \geq -1, j \in J_2\}$ and let I_q denote the set of indices defining the disjunction variables for B_q. Because we have the righthand side equal to $\pi^k_0(x, \omega)$ for all $k \in \kappa(\tau), \tau \in B_q$, the cuts used in the second-stage are facets of $Y(x) \cap (\{y_j \leq 0\} \cup \{y_j \geq 1\})_{j \in I_q}$. Since the feasible set of the second-stage can be described as a facial disjunctive set, the sequential convexification process yields the

convex hull of the set as proved by [2]. Therefore, after finitely many divisions t, the branch-and-bound process finds a node with either $V_q - v_q < \epsilon$ or an infeasible nodal master program (12). ■

Theorem 2 *Suppose that assumptions (A1-A3) hold. The proposed D^2-CBAC algorithm terminates with an optimal solution to problem (1-2) after finitely many steps.*

Proof In the proposed D^2-CBAC algorithm branching can only be done finitely many times. This follows from the fact that there are finitely many disjunction variables in the second-stage, leading to finitely many righthand sides $\{\pi_0^k(x,\omega)\}_{k\in\Theta_K}$ for some $K < \infty$ for all $\omega \in \Omega$. Consequently, there are finitely many divisions of the continuous first-stage feasible set to consider. Then by Lemma 1 each node q will be fathomed in the course of the algorithm. The optimality of the solution follows from the validity of the lower and upper bounding procedures used.■

Some comments on the critical operations of the D^2-CBAC algorithm are now in order. The *branching* part of the algorithm requires that the multipliers that define $\pi_0^k(x,\omega)$ and $\pi_c^k(x,\omega)$ for all k and $\omega \in \Omega$ be recorded. This data may be large and therefore, writing them to a file may be necessary for computer memory considerations. In Step 1 of the algorithm a nodal problem must be *selected* from the list $\mathcal{L}$. We propose selecting a nodal problem with the least lower bound v_q. That is, selecting a problem $P_{\bar{q}}$ such that $v_{\bar{q}} = \min_{q\in\mathcal{L}}\{v_q\}$. This follows the *best-node first strategy* and leads to a bound improving selection operation as required for the convergence of a branch-and-bound algorithm [7].

Since the D^2 cuts generated at a given node are not in general valid for all nodes in the branch-and-bound tree, care must be taken to curtail the proliferation of these cuts. We suggest avoiding storing all the cuts explicitly for each node. Instead, the cuts can be stored in some data structure that contains each cut with the corresponding algorithmic iteration and node number at which the cut was generated. Thus each cut would be recorded only once and used for each nodal problem as required. When a node is *fathomed* (by infeasibility, bound or optimality) all the D^2 cuts that were generated at that node can be deleted from the list. This also applies to the branching constraints and optimality cuts generated at a node and added to the master program. Next we provide an example to illustrate the proposed algorithm.

5 Illustrative Numerical Example

Consider the following simple instance of an SMIP in extensive form:

$$\begin{aligned}
\min\ & -2x - 2y_1 - 2y_2 \\
\text{s.t. } & -x \geq -2 \\
& -5x - 6y_1 \geq -10 \\
& -5x - 6y_2 \geq -12 \\
& -y_1 \geq -1 \\
& -y_2 \geq -1 \\
& x \geq 0, y_1, y_2 \text{ binary.}
\end{aligned}$$

Decomposing it as a two-stage SMIP we have:

$$\begin{aligned}
\min\ & -2x + E[f(x,\omega)] \\
\text{s.t. } & -x \geq -2 \\
& x \geq 0,
\end{aligned}$$

where,

$$\begin{aligned}
f(x,\omega) = \min\ & -4y \\
\text{s.t. } & -6y \geq r(\omega) + 5x \\
& y \text{ binary.}
\end{aligned}$$

The second-stage has two scenarios, each having a probability of 0.5 with $\omega = \omega_1$ having $r(\omega_1) = -10$, and $\omega = \omega_2$ with $r(\omega_2) = -12$. Note that the problem satisfies assumptions (A1-A3). In this instance we have the following data: $c = -2,\ A = [-1],\ b = [-2],\ q = [-4],\ W = [-6],\ T(\omega_1) = T(\omega_1) = [-5],\ r(\omega_1) = -10$ and $r(\omega_2) = -12$. Note that the lower bound on the expected value of the second stage problem is $L = 0.5 * -4 + 0.5 * -4 = -4$. We can now apply the D^2-CBAC algorithm as follows.

Iteration 1 ($k = 1$)

Step 0: Initialization
The algorithm is initialized with the following root node (Node 0) problem P_0 whose master program is:

$$\begin{aligned}
\min\ & -2x + \eta \\
\text{s.t. } & -x \geq -2 \\
& x, \eta \geq 0.
\end{aligned}$$

Set $W^1 = W,\ T^1(\omega) = T(\omega)$, and $r^1(\omega) = T(\omega)$ for both scenarios and initialize the list open nodal problems $\mathcal{L} \leftarrow \mathcal{L} \cup \{P_0\}$. The global upper and lower bounds are initialized as $V = \infty$ and $v = -\infty$, respectively.

Step 1: Termination

List $\mathcal{L} \neq \emptyset$, so we select the initial nodal problem. The initial master program yields $x^1 = 2$ and $\eta = 0$. The nodal upper and lower bounds are $V_0 = \infty$ and $v_0 = -\infty$, respectively.

Step 2: Apply one iteration of the D^2 Algorithm to the Nodal Problem

Step (i)
For Step 1 of the algorithm we use $x^1 = 2$ and solve the LP relaxation of the second-stage subproblem for ω_1 and ω_2, which we shall call LP_1 and LP_2, respectively:

$$\begin{aligned} LP_1 : f_1(2, \omega_1) = \min \; & -4y \\ \text{s.t.} \; & -6y \geq 0 \\ & -y \geq -1 \\ & y \geq 0. \end{aligned}$$

and

$$\begin{aligned} LP_1 : f_2(2, \omega_2) = \min \; & -4y \\ \text{s.t.} \; & -6y \geq -2 \\ & -y \geq -1 \\ & y \geq 0. \end{aligned}$$

The optimal solution for LP_1 is $y(\omega_1) = 0$ and for LP_2 is $y(\omega_2) = 0.3333$

Step (ii)
Since $y(\omega_2)$ does not satisfy the binary restrictions, we choose y as the "disjunction" variable and create the disjunction $-y \geq 0$ or $y \geq 1$ for LP_2. We formulate the C^3-LP (problem 18 in [22]), which yields the vector π^1 for updating W^1 and the multipliers for forming the RHS-LP. Solving the C^3-LP yields $\pi^1 = -1, \lambda_{0,1}^\top = [0, 0], \lambda_{0,2} = 1, \lambda_{1,1}^\top = [0.25, 0]$, and $\lambda_{1,2} = 0.5$ as the optimal solution.

We obtain W^2 by appending π^1 to W^1: $W^2 = \begin{bmatrix} [W^1] \\ -1 \end{bmatrix}$. From the C^3-LP solution we obtain $\bar{\nu}_0^1(\omega_1) = 0$, $\bar{\nu}_1^1(\omega_1) = -2$, $\bar{\gamma}_0^1(\omega_1)^\top = [0]$, $\bar{\gamma}_1^1(\omega_1)^\top = [-1.25]$ and $\bar{\nu}_0^1(\omega_2) = 0$, $\bar{\nu}_1^1(\omega_2) = -2.5$, $\bar{\gamma}_0^1(\omega_2)^\top = [0]$, $\bar{\gamma}_1^1(\omega_2)^\top = [-1.25]$. In order to maintain $\pi_0^k(x, \omega) \geq 0$, we translate $\bar{\nu}_0^1(\omega)$ and $\bar{\nu}_1^1(\omega)$ by $+2$ and $+2.5$, respectively. We then have:

$$\pi_0^1(x, \omega_1) = \min\{2 - 0x, 0 + 1.25x\}$$

and

$$\pi_0^1(x, \omega_2) = \min\{2.5 - 0x, 0 + 1.25x\}.$$

Using the above data we formulate the RHS-LP (problem 19 in [22]) for both ω_1 and ω_2. The optimal solution for ω_1 is $\delta(\omega_1) = 0$, $\sigma_0(\omega_1) = 0.55556$, $\sigma(\omega_1) = -0.555556$ and for ω_2 is $\delta(\omega_1) = 0$, $\sigma_0(\omega_1) = 0.5$, $\sigma(\omega_1) = -0.625$.

We therefore have $\nu^2(\omega_1) = 0$, $\gamma^2(\omega_1) = 1$, $\nu^2(\omega_2) = 0$ and $\gamma^3(\omega_1) = 1.25$. Translating back we obtain

$$\pi_c^1(x, \omega_1) = -2 + x$$

and

$$\pi_c^1(x, \omega_2) = -2.5 + 1.25x.$$

Note that since $x^1 = 2$, $x^1 \in vert(X)$, we have that $\pi_0^1(2, \omega_1) = \pi_c^1(2, \omega_1) = 2$ and $\pi_0^1(2, \omega_2) = \pi_c^1(2, \omega_2) = 2.5$. Based on the RHS-LP solution we make the following updates: $r^2(\omega_1) = \begin{bmatrix} [r^1(\omega_1)] \\ -2 \end{bmatrix}$, $T^2(\omega_1) = \begin{bmatrix} [T^1(\omega_1)] \\ -1 \end{bmatrix}$. Similarly we update $r^2(\omega_2) = \begin{bmatrix} [r^1(\omega_2)] \\ -2.5 \end{bmatrix}$, $T^2(\omega_2) = \begin{bmatrix} [T^1(\omega_2)] \\ -1.25 \end{bmatrix}$. This completes Step 2 of the algorithm.

Step (*iii*)

Re-optimizing the updated scenario subproblems we obtain $y(\omega_1) = 0$ with $f_1(2, \omega_1) = 0$, and $y(\omega_2) = 0$ with $f_2(2, \omega_2) = 0$. Note that the fractional solution for ω_2 has been cut off by the D^2 cut. The dual solutions are $d^\top(\omega_1) = d^\top(\omega_2) = [0, 0, 4]$. Since both scenarios satisfy binary requirements on y we have an incumbent solution $x = 2$ and we update the upper bound $V_0 = \min\{V_0, -2 * 2 + 0.5 * (0) + 0.5 * (0) + 4\} = 0$.

Step (*iv*)

Using the dual solution for each scenario subproblem LP from Step (*iii*), we formulate the Benders-type optimality cuts. The resulting cuts are $-4x + \theta(\omega_1) \geq -8$ and $-5x + \theta(\omega_2) \geq -10$. Since the two scenarios are equally likely, the expected values associated with the cut coefficients yield $-4.5x + \theta \geq -9$. Since we assume that $\eta \geq 0$, we apply the translation $\eta = \theta + 4$ to get $-4.5x + \eta \geq -5$ as the optimality cut to add to the master program:

$$\begin{aligned} \min \ & -2x + \eta \\ \text{s.t.} \ & -x \qquad \geq -2 \\ & -4.5x + \eta \geq -5 \\ & x, \eta \geq 0. \end{aligned}$$

Solving the master program we get $x^2 = 1.111$, $\eta = 0$ and an objective value of -2.222. Therefore, the lower bound $v_0 = -2.222$. This completes this iteration of the D^2 algorithm. Since $V_0 > v_0$, $k \leftarrow 2$, and begin the next step of the D^2-CBAC algorithm. We note that $x^2 \notin vert(X)$ and so branching will be necessary.

Step 3: Branching

We now need to apply Proposition 1 with $x^2 = 1.111$ (use the translated functions $\pi_0^1(x, \omega)$ and $\pi_c^1(x, \omega)$):

For ω_1 we have:

$$\begin{aligned}\pi_0^1(1.111,\omega_1) &= \min\{2-0*1.5, 0+1.25*1.111\}\\ &= \min\{2, 1.389\}\\ &= 1.389\end{aligned}$$

$$\begin{aligned}\pi_c^1(1.111,\omega_1) &= 0+1*1.111\\ &= 1.111\end{aligned}$$

Thus

$$\begin{aligned}p_\omega(\pi_0^1(1.111,\omega_1)-\pi_c^1(1.111,\omega_1)) &= 0.5*(1.389-1.111)\\ &= 0.1389.\end{aligned}$$

For ω_2 we have:

$$\begin{aligned}\pi_0^1(1.111,\omega_2) &= \min\{2.5-0*1.111, 0+1.25*1.111\}\\ &= \min\{2.5, 1.389\}\\ &= 1.389\end{aligned}$$

$$\begin{aligned}\pi_c^1(1.111,\omega_2) &= 0+1.25*1.111\\ &= 1.389\end{aligned}$$

Thus

$$\begin{aligned}p_\omega(\pi_0^1(1.111,\omega_2)-\pi_c^1(1.111,\omega_2)) &= 0.5*(1.389-1.389)\\ &= 0.\end{aligned}$$

In this case we select $(\varpi,\kappa) = (\omega_1, 1)$ since $0.1389 > 0$. The branching constraint is therefore,

$$\begin{aligned}&\bar{\nu}_0^1(\omega_1)-\bar{\gamma}_0^1(\omega_1)x \geq \bar{\nu}_1^1(\omega_1)-\bar{\gamma}_1^1(\omega_1)x\\ \Longrightarrow\ &2-0x \geq 0+1.25x\\ \Longrightarrow\ &-x \geq -1.6\end{aligned}$$

So now we have for the first branch $-x \geq -1.6$ with the corresponding D^2 cut in the subproblem updated to $\pi^1 y \geq \bar{\nu}_1^1(\omega_1)-\bar{\gamma}_1^1(\omega_1)x \implies -1y \geq -2+1.25x$. For the second branch we have $x \geq 1.6$ with the corresponding D^2 cut updated to $\pi^1 y \geq \bar{\nu}_1^1(\omega_1)-\bar{\gamma}_1^1(\omega_1)x \implies -1y \geq 0-0x$ or $-y \geq 0$. Note that for this branch y is fixed at 0. We now create two nodal problems (P_1 and P_2) whose master programs are as follows.

Node 1 Master Program:

$$\begin{aligned}\min\ & -2x+\eta\\ \text{s.t.}\ & -x \geq -2\\ & -4.5x+\eta \geq -5\\ & -x \geq -1.6\\ & x,\eta \geq 0.\end{aligned}$$

with optimal solution $x^2 = 1.111$, $\eta = 0$ and an objective value of -2.222. Therefore, the lower bound $v_1 = -2.222$.

Node 2 Master Program:

$$\begin{aligned} \min\ & -2x + \eta \\ \text{s.t.}\ & -x \geq -2 \\ & -4.5x + \eta \geq -5 \\ & x \geq 1.6 \\ & x, \eta \geq 0. \end{aligned}$$

with optimal solution $x^2 = 1.6$, $\eta = 2.2$ and an objective value of -1. Therefore, the lower bound $v_2 = -1$. Set $\mathcal{L} \leftarrow \mathcal{L} \cup \{P_1, P_2\}$ to the list of unexplored problems $\mathcal{L}$.

Step 1: Termination

List $\mathcal{L} \neq \emptyset$. Since $v_1 < v_2$ we *s*elect P_1 and set $\mathcal{L} \leftarrow \mathcal{L} \setminus \{P_1\}$.

Step 2: Apply one iteration of the D^2 Algorithm to the Nodal Problem

Starting with $x^2 = 1.111$ we perform the D^2-CBAC and and obtain the results summarized in the branch-and-bound tree shown in Figure 3 in the Appendix. The algorithm explores a total of five nodes, with node 3 fathomed by optimality, while nodes 2 and 4 are fathomed by bound. Then list $\mathcal{L} = \emptyset$ and the algorithm terminates with the optimal solution $x = 0.8$, $y(\omega_1) = 1$, $y(\omega_2) = 1$ and an objective function value of -5.6.

6 Conclusion

This paper presents a novel D^2 with branch-and-cut method for two-stage SMIP with continuous first-stage variables or D^2-CBAC. The D^2-CBAC algorithm is derived based on the D^2 algorithm, a pure cutting-plane method for SMIP which requires the first-stage solutions to be extreme points of the first-stage feasible set. This requirement is not necessary in the D^2-CBAC method, whose novelty derives from the fact that branching is done on the first-stage continuous domain. In essence, the branch-and-bound process is guided by the disjunction variables in the second-stage. Finite convergence of the algorithm for mixed-binary second-stage is established and a numerical example to illustrate the new method is given.

Acknowledgments This research was funded by grants from the OR Program (DMII-9978780), and the Next Generation Software Program (CISE-9975050) of the National Science Foundation.

References

1. Ahmed, S., Tawarmalani,M., Sahinidis, N.V.: A finite branch and bound algorithm for two-stage stochastic integer programs, Math. Programming, **100**, 355–377 (2004).

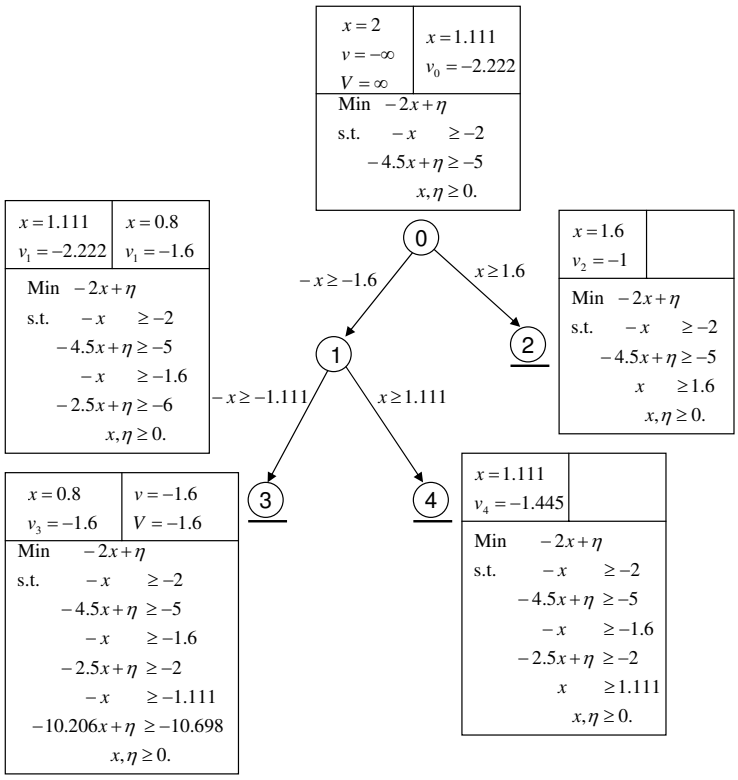


Fig. 3 Branch-and-bound tree for the illustrative example

2. Balas, E.: Disjunctive Programming, Annals of Discrete Mathematics, **5**, 3–51, 1979.
3. Benders, J. F.: Partitioning procedures for solving mixed-variable programming problems. *Numerische Mathematik* **4**, 238–252 (1962).
4. Birge, J. R., Louveaux,F. V.: Introduction to Stochastic Programming, Springer, New York (1997).
5. Blair, C., Jeroslow, R.: The value function of an integer program, Mathematical Programming, **23**, 237–273 (1982).
6. Carøe, C.C.: Decomposition in stochastic integer programming, Ph.D. thesis, Dept. of Operations Research, University of Copenhagen, Denmark (1998).
7. Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches, Springer-Verlag, Berlin, 3rd Ed. (1996).
8. Klein Haneveld, W.K., van der Vlerk,M.H.: Stochastic integer programming: general models and algorithms, Annals of Operations Research, **85**, 39–57 (1999).
9. Laporte, G., Louveaux, F.V.: The integer L-shaped method for stochastic integer programs with complete recourse, Operations Research Letters, **13**, 133–142 (1993).
10. Ntaimo, L.: Decomposition Algorithms for Stochastic Combinatorial Optimization: Computational Experiments and Extensions, Ph.D. Dissertation, University of Arizona, Tucson, USA (2004).
11. Ntaimo, L., Sen, S.: The million-variable 'march' for stochastic combinatorial optimization, J. of Global Optimization, **32(3)**, 385–400 (2004).
12. Schultz, R.: Continuity properties of expectation functions in stochastic integer programming, Math. of Operations Research, **18**, 578–589 (1993).

13. Schultz, R., Stougie, L., van der Vlerk, M.H.: Two-stage stochastic integer programming: A survey, Statistica Neerlandica, **50**,404–416 (1996)
14. Sen, S., Sherali,H. D.: Nondifferentiable reverse convex programs and facetial cuts via a disjunctive characterization, Math. Programming, **37**, 169–183 (1987).
15. Sherali,H.D., Adams, W.P.: A hierarchy of relaxations between the continuous and convex hull representations for 0-1 programming problems, SIAM J. on Discrete Mathematics, **3**, 411–430, (1990)
16. Sherali,H.D., Adams, W.P.: A hierarchy of relaxations and convex hull characterizations for zero-one programming problems, Discrete Applied Mathematics, **52(1)**, 83–106 (1994).
17. Sherali,H.D., Adams, W.P.: A Reformulation-Linearization Technique for Solving Discrete and Continuous Nonconvex Problems, Kluwer, Boston, MA (1999).
18. Sherali,H.D., Fraticelli, B.M.P.: A modification of Benders' decomposition algorithm for discrete subproblems: an approach for stochastic programs with integer recourse, J. of Global Optimization, **22**, 319–342 (2002).
19. Sherali, H.D., Zhu, X.: On solving discrete two-stage stochastic programs having mixed-integer first- and second-stage variables, Math. Programming, accepted (2005).
20. Sen, S., Higle,J. L., Ntaimo, L.: A summary and illustration of disjunctive decomposition with set convexification. D. L. Woodruff, ed., Stochastic Integer Programming and Network Interdiction Models, Kluwer Academic Press, Dordrecht, The Netherlands. Chapter 6, 105–123 (2002).
21. Sen, S.: Algorithms for stochastic mixed-integer programming models. In: K. Aardal, G.L. Nemhauser, R. Weismantel (eds.) Handbooks in Operations Research and Management Science, 12: Discrete Optimization, Dordrecht, North-Holland, Chapter 9.
22. Sen, S., Higle, J.L.: The C3 Theorem and a D2 Algorithm for Large Scale Stochastic Mixed-Integer Programming: Set Convexification. Math. Programming, **104(1)** 1–20 (2005).
23. Sen, S., Sherali, H.D.: Decomposition with Branch-and-Cut Approaches for Two Stage Stochastic Mixed-Integer Programming, Math. Programming, **106(2)**, 203–223 (2006).